EPM/RT-94/27

A Hierarchical Approach to Aggregating Software Systems Evaluation Data

Germinal Boloix
Pierre N. Robillard

Département de génie électrique
et génie informatique

École Polytechnique de Montréal
Novembre 1994

Dépôt légal, novembre 1993
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Pour se procurer une copie de ce document, s'adresser:

Les Éditions de l'École Polytechnique
École Polytechnique de Montréal
Case postale 6079, succ. Centre-ville
Montréal, (Québec) H3C 3A7
Téléphone: (514) 340-4473
Télécopie: (514) 340-3734

Compter 0.10 $ par page et ajouter 3,00 $ pour la couverture, les frais de poste et la manutention. Régler en dollars canadiens par chèque ou mandat-poste au nom de l'École Polytechnique de Montréal.

Nous n'honorerons que les commandes accompagnées d'un paiement, sauf s'il y a eu entente préalable dans le cas d'établissements d'enseignement, de sociétés ou d'organismes canadiens.

A Hierarchical Approach to Aggregating Software Systems Evaluation Data

Germinal Boloix, Pierre N. Robillard
16 November 1994
Ecole Polytechnique de Montréal
Département de Génie Electrique et de Génie Informatique
C.P. 6079 succ Centre Ville
Montréal, Québec H3C 3A7
boloix@rgl.polymtl.ca

Abstract

Presenting information about software systems to high-level managers is a challenge. Decision-makers do not need the myriad of detail available at lower levels. A metrics program has to address the full range of information needs required by software developers, operators and users, up to higher-level ranks in the organization. A hierarchical approach to aggregating software systems evaluation data is proposed which builds upon a software systems evaluation framework structured in a hierarchy representing three dimensions (i.e., project, system, environment). The approach defines the relative importance of software systems attributes through pairwise comparisons at each level of the hierarchy. The aggregated data proceeds from low-level factors to high-level dimensions. Examples of evaluations demonstrate the potential of the approach.

Keywords: software systems evaluation framework, software systems evaluation, software metrics, Total Quality Awards, Analytic Hierarchy Process

1.0 Introduction

Information about software systems, in the form of software metrics, provides supporting aids for more accurate estimation of project milestones, the establishment of requirements during the initial stages of development, the monitoring of project progress and the evaluation of project results. Measures can be used to set goals for productivity and quality and to establish a baseline against which improvements can be compared. The amount of available information about software systems and its interrelationships is so extensive that simplifying mechanisms are required to summarize several views in combination. Several metrics viewed in concert with other metrics are more useful than isolated metrics.

High-level managers require information that has to be abstracted from a myriad of detail, but at the same time, detail has to be available. Hot-spot data, at high levels of abstraction, require facilities to determine their seriousness and identify the causes. Different points of view, technical or managerial, can utilize the information according to their particular needs. An evaluation mechanism that addresses a wider range of needs is more attractive than one which is highly focused on a small set of elements or points of view. Organizational success depends on the quality of products and services and the ability to respond efficiently to customers; software systems play a major role in the accomplishment of these organizational objectives.

Collective analyse of software metrics have been proposed which aggregate available software metrics. Card and Glass describe a complexity metric that is a composite of different aspects of complexity [CG90]. Munson and Khoshgoftaar apply a similar technique to aid in project management decisions [MK90]. Pfleeger et al. propose a graphical representation, a variation of Kiviat diagrams called 'multiple metrics graphs', which considers the relative importance of the metrics [PFR92]. Total quality evaluation approaches aggregate corporate data using points and percentages [MB94, EQ94].

A hierarchical approach to aggregating evaluation data is proposed in this paper. It follows a software systems evaluation framework which has been developed in top-down fashion. Lower-level assessments of factors are aggregated into high-level dimensions in the framework, according to the relative importance of the factors. The approach represents a convenient way of documenting the criteria defined by each organization to evaluate its software systems. The originality of the approach lies in the clustering of related

factors into common perspectives that support the points of view of evaluators. Information about software systems is useful in identifying sets of similar systems and the range of projects undertaken by an organization.

The article is organized as follows. Section 2 presents examples using the framework for evaluation. Section 3 describes the hierarchical approach to software systems evaluation with some examples. Section 4 introduces other quantitative approaches taken from current literature. Section 5 gives some conclusions and describes further research already under way. In the appendix, details about the software systems evaluation framework itself and a summary of corporate quality awards are described.

2.0 Software systems evaluation framework

A software systems evaluation framework which facilitates the selection of a basic set of attributes to characterize the important dimensions of software systems has been proposed [BR94]. A top-down approach identifies the main dimensions and factors in a hierarchy, and has the advantage of providing an evaluation tool which can be subject to formal (mathematical) analysis. The framework assigns maturity levels (i.e., basic, intermediate, advanced) to each factor in the framework.

2.1 Model dimensions

The framework is organized in three dimensions, according to three different viewpoints (i.e., those of the developer, the operator and the user). The information about software systems being retained concerns the software system, its production process and its impact on the organization. software systems are more than its internal characteristics and the production process which creates it, software systems provides a service to users. The viewpoints represented in the framework depict the project, the system and the environment. These viewpoints are also interpreted according to the software life-cycle: before the software system is operational (i.e., development or enhancement), during the operation of the system (i.e., installation and operation), and after results of operating the system have become available (i.e., post-evaluation stage).

The three dimensions in the framework are interconnected. A software *system* may be *subject_to* several evolution *projects* during its lifetime: initial development and

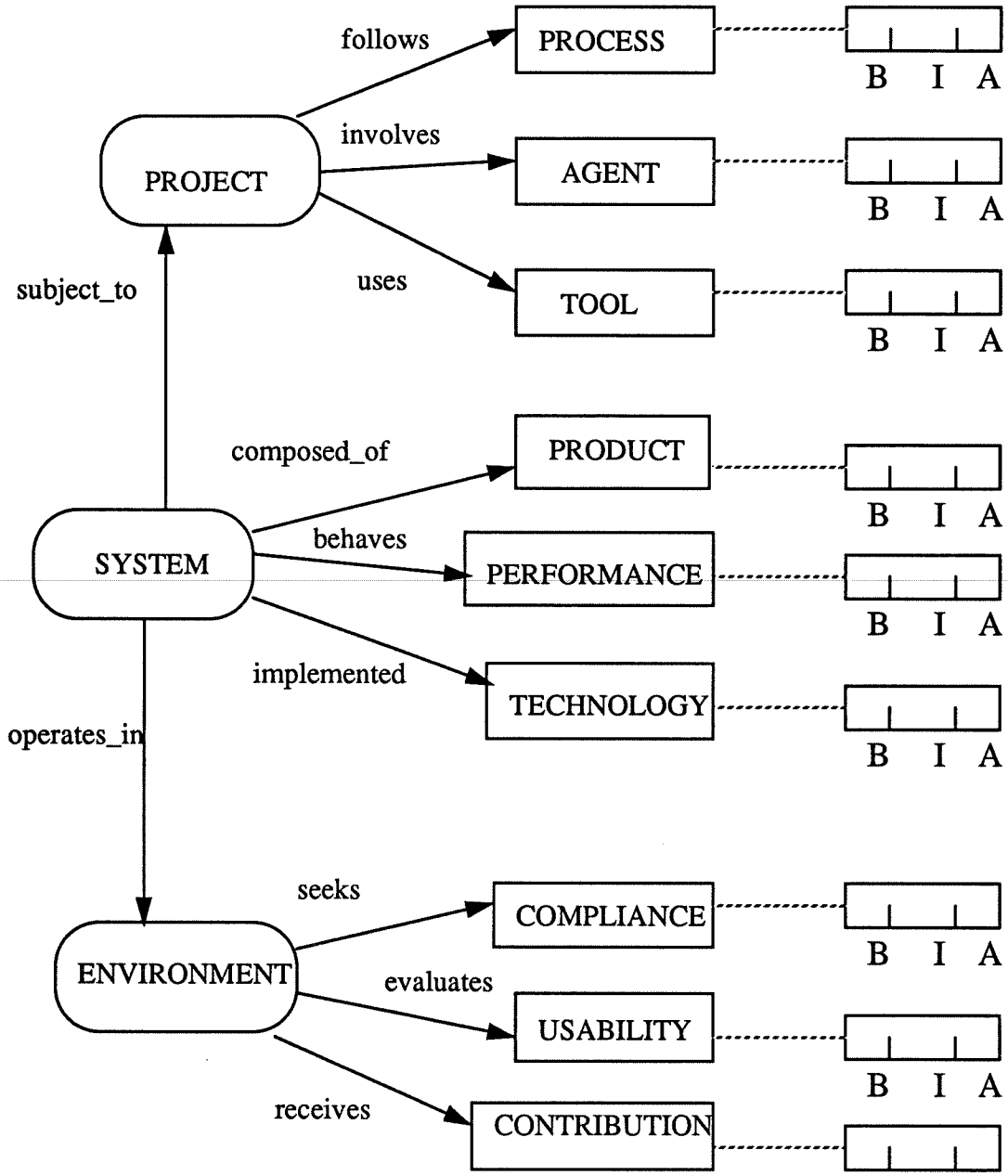
enhanced versions. A system *operates_in* an organizational *environment*: users interface with the system and services are provided by the system.

Each dimension of the framework is decomposed into its important factors. Thus, the project dimension recognizes characteristics of the process, the agents and the tools utilized during software systems production. The system dimension characterizes the internals of the product: product, technology and performance factors. Finally, the environment dimension addresses the externals of the system: compliance with requirements, usability of the system and contribution of the system to the organization.

Figure 1 presents an entity-relationship diagram which further decomposes the framework into factors. A project *follows* a process, it *involves* some agents and *uses* some tools. The system is *composed_of* products, it *behaves* at some performance level and it is *implemented* in a particular technology. The environment *seeks* compliance with system requirements, it *evaluates* the usability of the system from the user's perspective and it *receives* a contribution or benefit from the operation of the system.

A third level of decomposition is introduced to categorize factors in the framework. Categories are useful for classifying information about software systems from a maturity perspective. To keep categories simple, only three ratings have been identified: basic (B), intermediate (I) and advanced (A). A basic category indicates the lowest maturity rating for a particular factor. An intermediate category may indicate a nominal rating or a standard in the industry, whereas an advanced category identifies a higher maturity rating which demonstrates excellence. Details of the framework are described in the appendix.

FIGURE 1. Categories in the framework



2.2 CASE Tool Evaluation

To understand the steps of software systems evaluation using the framework, examples of evaluations are presented. The evaluations reflect the'actual rating' for a system rather than its 'goal rating'. These examples correspond to software tools developed over the last three years by the same team of developers. Once evaluators have been introduced to the framework, each factor is analyzed and one category is chosen by concensus. Ideally, evaluators should represent the three points of view of the framework. We present a summary of the results in the following tables.

Structure editor

A structure editor which assists programmers during algorithm design and code generation was evaluated following the framework. This tool is the result of many years of research in software engineering. We present the results of the evaluation for the latest version of this tool. At the project dimension level, it is possible to identify, for agents (B), a small team of people following a semi-formal project organization; the process (I) model is standard, following structured techniques, and involving reuse of software products; commercial tools (I) are available for development and project control, including workbenches but not integrated software development environments.

The system dimension portrays the product (B) as a relatively small software system with a low degree of complexity; performance (I) considerations indicate a system with intermediate requirements in terms of reliability and efficiency; technology (I) makes an intermediate target language, in this case C, available in an open-system environment. The environment dimension indicates satisfaction from the user's perspective. Compliance (A) with requirements is high, as this version of the software system is the result of several years of evolution. Usability (A) is advanced, as the software system is targeted for use by professional programmers. The contribution (I) to the users is intermediate, as this software system saves user time during algorithm development and maintenance.

Static analyzer

A software quality assessment tool which analyzes source code and translates it into a formal representation of the source program, and which is programming-language-inde-

pendent, was also evaluated. A precise computation of many traditional, and innovative metrics is performed using this tool. See Table 1 for the results of the evaluation according to the framework.

Table 1: Tool categories

	Structure Editor	Static Analyzer
Process	I	I
Agent	B	B
Tool	I	I
Performance	I	A
Product	B	B
Technology	I	A
Compliance	A	I
Usability	A	I
Contribution	I	I

(B: basic, I: intermediate, A: advanced)

The project dimension in the static analyser is similar to that in the structure editor. The process (I) model, agents (B) and tools (I) were basically the same. Minor changes tended to improve the management of the process for the static analyser. A record of changes in the process, introduced during the development of this tool, was kept. The individuals working on the two projects were different, their tasks overlapping only at the supervisory level, but for the purposes of the framework they were equivalent. In terms of the system dimension, the static analyzer is equivalent to the structure editor as far as the product (B) factor is concerned. Being a source code analyzer, the tool was developed with more stringent requirements in efficiency (A). In terms of technology (A), this tool was built using C++ and open-system technology. As for the environment dimension, there is a slight variation in the tools. The static analyzer is a research tool, thus a user interface oriented towards experts which do not require a sophisticated interface, thus usability (I) is intermediate. Compliance (I) is intermediate as this is a tool in evolution. Contribution (I) is intermediate as the tool benefits a reduced community.

Rule-based Approach

The assignment of categories at the dimension level is achieved by grouping its factor categories. The assignment of categories follows a rule-based approach. We present this approach only for purposes of comparison with the quantitative approach. Rules have the following appearance:

IF *Agent* = 'basic' AND *Process* = 'intermediate' AND *Tool* = 'intermediate'

THEN *Project* = 'low'

Table 2 shows the assignment of categories at the project dimension level. This assignment is empirical, by analyzing the alternative groupings of factor categories. The assignment is done by expert opinion, according to the importance of each factor. The rule-based approach is a symbolic-assignment alternative rather than a quantitative one. An equivalent table, not necessarily identical, can be built for each of the other dimensions (i.e., system and environment).

Table 2: Project category assignment

Low	Medium	High
B - B - B	I - I - I	A - A - A
B - I - B	B - A - I	A - I - A
B - B - I	I - B - A	I - A - A
I - B - B	A - B - I	A - A - I
I - B - I	I - A - B	A - I - I
B - I - I	B - I - A	I - A - I
I - I - B	A - I - B	I - I - A
B - B - A	A - A - B	
B - A - B	B - A - A	
A - B - B	A - B - A	

(Process - Agent - Tool)

The rational to assign alternative groupings of factor categories at the project dimension level is as follows. A category of 'Low' at the project level is assigned when two or

three evaluations of factors are 'B' (basic) or one evaluation is 'B' (basic) and the other two are 'I' (intermediate). The assignment of 'High' is done when two or three evaluations of factors are 'A' (advanced) or there is one factor 'A' and the other two are 'I' (intermediate). The assignment of 'Medium' is done for the rest of alternatives.

Table 3: Comparison of tools

	Structure Editor	Static Analyzer
PROJECT	Low	Low
SYSTEM	Low	Medium
ENVIRONMENT	High	Medium

Table 3 shows the overall classification of the tools according to each dimension. It is clear from these results that the organization developed the tools through equivalent projects. Differences in the evaluation appear at the system and environment dimension levels. At the system dimension level, one tool has higher performance than the other. At the environment dimension level, one tool provides more support from the user's perspective.

Experiments evaluating software systems according to this framework demonstrate that an evaluation requires a short time to complete, an average of 1.5 hours per system is expected. Categories are presented in a way that minimizes the effect of subjectivity in the selection process, and participants interact with interviewers to choose one option.

3.0 Hierarchical approach

The rule-based approach to aggregating factors into dimensions is defined empirically by grouping alternative factor evaluations in an ordinal scale. We propose a hierarchical approach to aggregate data from low-level factors into high-level dimensions which utilizes a ratio scale. The approach benefits from the hierarchical structure of the framework to identify each level and its interrelationships. Saaty [S80] proposed a decision model to select among alternatives, and this has been modified for the evaluation of systems.

A ranking method comparing the elements in the framework pairwise is suggested. To determine the importance among the elements in the framework, expert opinion is

required. Managers, technicians and users are clear candidates for interviewing. The advantage of defining the importance between the elements in the framework is better documentation, which is available for future reference. The relative importance of elements is an organizational issue, each organization can establish its own parameters. The choice of the relative importance of elements in the framework is independent of the choice of factor categories during an evaluation of a software system.

3.1 Importance in the framework

To determine the degree of importance of the elements in the framework, a pairwise comparison of elements at the same level in the hierarchy is required. A scale of 1 to 9 is used to determine the relative importance of each pair of elements: A and B are equally important, insert 1, A is weakly more important than B, insert 3, A is strongly more important than B, insert 5, A is demonstrably or very strongly more important than B, insert 7, A is clearly more important than B, insert 9. Other values such as 2, 4, 6 or 8 could also be utilized to define intermediate ratings.

Let us apply the method to the elements in the framework. Expert opinion was gathered from personnel involved in management of software tool development, producers of software tools and users of the tools. The following matrices show the assessment of importance following the levels of the framework. These matrices are not suggested as a norm, but as an example of how to determine the relative importance of elements.

Matrix 1: Importance of the dimensions in the framework

Framework	PROJECT	SYSTEM	ENVIRONMENT
PROJECT	1	1/3	1/5
SYSTEM	3	1	1/3
ENVIRONMENT	5	3	1

Matrix 1 shows that the ranking of factors is environment first, followed by system, then project (i.e., 5 to 3 to 1). The environment represents the usage and contribution to users, and thus has greater importance than the characteristics of the system. Note that reciprocals are used to define the rest of this values in the matrix; it is not mandatory to

enter a reciprocal, but it is generally rational to do so. The values in the diagonal are predetermined to 1, since, for example, a project has the same ranking as itself.

Matrix 2: Importance at the project dimension level

PROJECT	Agent	Process	Tool
Agent	1	5	7
Process	1/5	1	3
Tool	1/7	1/3	1

From the project point of view, agents are first, then process and finally tools. Agents are the more important resource since they define the success of projects. The process is important in that it guides the sequence of activities, and finally the tools represent a productivity enhancer.

Matrix 3: Importance at the system dimension level

SYSTEM	Product	Performance	Technology
Product	1	3	5
Performance	1/3	1	5
Technology	1/5	1/5	1

From the system point of view, product is first, followed by performance, then technology. Characteristics of the product determine maintenance problems. Perfomance considerations determine dynamic attributes and the sophistication of the technology defines the level of mastery by operators.

Matrix 4: Importance at the environment dimension

ENVIRONMENT	Compliance	Usability	Contribution
Compliance	1	3	1/5
Usability	1/3	1	1/5
Contribution	5	5	1

From the environment dimension perspective, contribution is first, followed by compliance, then usability. Contribution is related to benefits to users and to the organization. Compliance signifies conformance to user requirements, and usability reflects to interface considerations.

Let us present the last level of the hierarchy which corresponds to the relative importance of the maturity ratings of the categories (i.e., basic, intermediate, advanced). Only the agent's point of view is depicted in Matrix 5, and similar tables can be obtained for the rest of factors in the framework. The order from advanced to basic is 5, 3, 1.

Matrix 5: Importance of maturity ratings

Agent	Basic	Intermediate	Advanced
Basic	1	1/3	1/5
Intermediate	3	1	1/3
Advanced	5	3	1

Each matrix generates an eigenvector (computed by normalizing values per column and averaging per row) which ranks the relative importance of the elements at each level of the framework. Using the resultant eigenvector of maturity ratings as a weighting factor, each factor of the evaluated software system is weighted. Results are composed level by level to get a global appraisal of the software system. We show the stages of the process below.

For each matrix, its eigenvector is computed. Here we show the results for Matrix 1.

Eigenvector for Matrix 1:

Framework	PROJECT	SYSTEM	ENVIRONMENT	Eigenvector(M1)
PROJECT	(1) / [9]	(1/3) / [13/3]	(1/5) / [23/15]	0.1062
SYSTEM	(3) / [9]	(1) / [13/3]	(1/3) / [23/15]	0.2605
ENVIRONMENT	(5) / [9]	(3) / [13/3]	(1) / [23/15]	0.6333
normalized by	9	13/3	23/15	

Eigenvectors of matrices 2, 3 and 4 are weighted with the category assigned: basic (.1062), intermediate (.2605) and advanced (.6333), using one of the elements of Eigen-

vector(M5) as a multiplier and producing a Weighted-Eigenvector. Finally, the resultant Weighted-Eigenvectors from matrices 2, 3 and 4 are placed as rows in a matrix which is multiplied by Eigenvector(M1). Results portray the aggregated evaluation for each dimension level in Eval-vector. To get a global evaluation for the software system, the resultant Eval-vector is multiplied by Eigenvector(M1) to give the Global-value. Results are normalized by the maximum evaluation.

$$\begin{vmatrix} \text{Weighted-Eigenvector(M2)} \\ \text{Weighted-Eigenvector(M3)} \\ \text{Weighted-Eigenvector(M4)} \end{vmatrix} \text{ X } \begin{vmatrix} \text{E} \\ \text{i} \\ \text{g} \\ \text{v} \\ \text{(M1)} \end{vmatrix} = \begin{vmatrix} \text{Eval-vector} \end{vmatrix}$$

$$\begin{vmatrix} \text{Eval-vector} \end{vmatrix} \text{ X } \begin{vmatrix} \text{E} \\ \text{i} \\ \text{g} \\ \text{v} \\ \text{(M1)} \end{vmatrix} = \text{Global-value}$$

3.2 Hierarchical tool evaluation

Table 4 shows the results of the hierarchical evaluation for the tools presented in the last section. In parentheses are the results of the rule-based appraisal shown before.

Table 4: Hierarchical evaluation

	Structure Editor	Static Analyzer
PROJECT	.24 (Low)	.24 (Low)
SYSTEM	.34 (Low)	.73 (Medium)
ENVIRONMENT	.48 (High)	.41 (Medium)

Comparing the hierarchical results and the rule-based evaluation presented in the last section, it is possible to establish that the hierarchical approach provides more understanding of the evaluation results. The rule-based approach, as shown in this paper, does not consider the relative importance of attributes, thus it gives a poor assignment. The hierarchical approach, on the other hand, provides a quantitative mechanism which presents the evaluation within a range (i.e., normalizing by the maximum). The merit of the

hierarchical approach is that it uses a ratio scale, whereas the rule-based approach uses an ordinal scale.

From Table 4, it is possible to identify a high rating at the system dimension level (.73) from the static analyzer, although the rule-based approach assigns an intermediate rating. Also, at the environment dimension level, an intermediate rating (.48) is computed for the structure editor, although the rule-based approach assigns an advanced rating.

Let us analyze the results by dimension. At the project dimension level there is no difference, since both tools were developed in similar project environments. At the system dimension level there is a clear difference between the tools, the static analyzer performing better. Finally, at the environment dimension level, the assessment is almost equivalent (0.48 versus 0.41), since the structure editor is a commercial tool with an improved user interface.

3.3 Information system evaluation

System evaluations were performed in an industrial setting. Project leaders, from a consulting firm, were interviewed to produce evaluations for systems recently developed for some client organizations. All profiles in the sample were different, indicating the sensitivity of the model. The following table present the evaluation profiles of those systems.

Information System Profiles

	1	2	3	4	5	6	7	8
Process	I	I	I	I	I	I	I	I
Agents	A	I	A	I	I	I	A	A
Tools	I	B	I	I	I	B	B	I
Product	A	I	B	I	A	I	B	B
Performance	A	I	A	B	A	I	A	A
Technology	I	I	I	B	A	I	I	I
Compliance	I	I	I	I	A	I	I	I
Usability	I	A	A	I	A	B	I	I
Contribution	I	A	I	I	I	A	I	A

From these profiles, and using the rule-based approach, it is possible to find out similarities among information systems. Analyzing the results by column, it is possible to identify systems possessing similar characteristics. Systems 2 and 6 are similar in all their factors except usability (advanced versus basic). Systems 1 and 3 are similar except for product and usability factors.

It also is possible to analyze, by row, each factor in the table to find similarities. For the process factor, all the systems were considered intermediate. The systems were developed for different organizations by the same consulting firm using the same process modeling approach, which explains why they occupy the same process factor category. For the compliance factor, all the information systems were intermediate, except one which was advanced (system 5). The rest of the factors were more varied for the rest of the systems.

Analyzing profiles by dimension, it is possible to identify systems sharing the same evaluation pattern, thus possessing similarities. At the project dimension level (process, agents and tools), and using Table 2, systems 1, 3 and 8 share the same project evaluation 'High' (I, A, I); systems 2 and 6 'Low' (I, I, B); systems 4 and 5 'Medium' (I, I, I). At the system dimension level (product, performance and technology), systems 3, 7 and 8 share the same evaluation 'Medium' (B,A,I) and systems 2 and 6 as well (I, I, I). At the contribution dimension level (compliance, usability and contribution), systems 1, 4 and 7 share the same evaluation 'Medium' (I, I, I) as well as system 6 (I,B,A); the other systems are categorized as 'High'.

The following table presents the results using the hierarchical approach. The table is organized by dimension and provides a global evaluation of the systems. The results are normalized to 1 for each dimension.

Hierarchical evaluations of information systems (normalized)

	1	2	3	4	5	6	7	8
Project	**.66**	**.28**	**.66**	**.33**	**.33**	**.28**	**.61**	**.66**
System	**.84**	**.38**	**.53**	**.23**	**1.00**	**.38**	**.53**	**.53**
Environment	**.38**	**.96**	**.45**	**.38**	**.48**	**.90**	**.38**	**.93**
Global Evaluation	**.53**	**.73**	**.49**	**.33**	**.60**	**.70**	**.43**	**.80**

Using quantitative results, it is possible to identify similarities much more easily than using the information system profiles. Systems 1, 3 and 8 are similar at the project dimension level. Systems 3, 7 and 8 are similar at the system dimension level. Systems 1, 4 and 7 are similar at the environment dimension level. Systems 2 and 6 have similar project and system dimensions, and are very close at the environment level (.96 to .90). Systems 3 and 8 have similar project and system dimensions, but are far apart at the environment level (.45 to .93). From the global evaluation, it is possible to identify similar overall projects, in this case systems 2 and 6 (.73 to .70).

These results indicate that a quantitative approach has definite advantages over a rule-based approach. Measurement provides more accuracy when comparing different assessments. The hierarchical approach provides a powerful abstraction mechanism to interpret the characteristics of the system at each level of the hierarchy. As the number of factors or levels in the hierarchy increases, the hierarchical approach becomes still more useful. The potential for applying this method is encouraging and we are convinced of the importance of this research.

4.0 Other approaches to aggregate evaluations

Multiple metrics graphs can be used to aggregate software metrics. This method uses a graphical representation based on Kiviat diagrams, determining the relative importance of metrics without considering levels of hierarchy. Also, the importance is not determined pairwise, but using weights for each metric.

Quality awards have been instituted which provide a method for quantifying the overall organization quality, and which primarily use point counting and percentage techniques. The awards we have analyzed are the Malcolm Baldrige Award and the European Quality Award.

4.1 Multiple Metrics Graphs

A multiple metrics graph is a Kiviat graph divided into unequal slices, so that the size of the slice represents the importance of the metric. Metrics are weighted by experts and the number of degrees in the arc are assigned to each slice. The goal of each metric is represented by an inside circle, the maximum is represented by an outer circle. A point is placed in each pie slice to represent the performance of the metric with respect to the goal.

A line from the center of the pie to the outer circle (through the middle of the arc forming the slice) represents the set of possible results; the closer to the center, the better the metric's performance. Hence, the smaller the area, the better the performance.

Next, lines are drawn from each of the representative points to the places where the goal arc meets the edges of the slice. In this way, the area of the resulting quadrilateral for each metric represents the degree to which goals are met: the smaller the area, the better. The polygon formed by uniting the three quadrilaterals represents the overall performance of the project or system. Below we show the Kiviat diagrams, which correspond to the degree of importance attached to the elements of the framework.

Let us now apply the multiple metrics graph approach to computing the surface area for the information systems evaluated. We show results at the dimension level, in order to be able to make a comparison with the hierarchical approach. The inside circle in Figure 2 represents the minimum evaluation for each dimension rather than a goal, and the outer circle represents the maximum evaluation. In this context, the larger the area, the better the evaluation. Surface results have been normalized to the maximum evaluation. The pointed line on the middle of each slice is used to measure the evaluation, from the internal circle to the outside circle. The same evaluation results of the hierarchical approach are assigned to each dimension (e.g., for system 1 project is .66, system is .84 and environment is .38, a normalized surface of .62 is obtained).

FIGURE 1. Area evaluation method

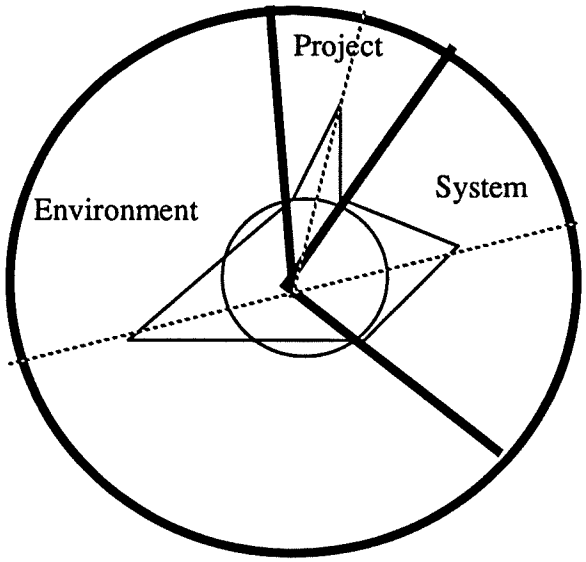


Figure 2: Surface evaluation method

Table I shows the results for each of the information systems evaluated before, using the multiple metrics graph method adapted to our needs. Results show the global area within the polygon, normalized by the maximum evaluation surface. System 2 receives the largest evaluation followed by system 8 and system 5. System 4 receives the smallest evaluation. At the end of this section an analysis of the results of the different methods will be presented.

Table 1: Multiple metrics graph evaluation

	1	2	3	4	5	6	7	8
Surface Evaluation	.62	.85	.54	.33	.71	.61	.49	.77

4.2 Quality awards

The software systems evaluation framework is related to quality awards because notions of total quality in evaluating software systems are important in both approaches. However, the framework was not developed to evaluate an overall organization. Our objective in this paper is to compare only the quantitative aspects of these approaches and not the framework content.

Malcolm Baldrige Award

This award institutes a method of points assigned per category. Each category contains items for evaluation, and there are more detailed criteria for each category. A percentage score is used to decide on the evaluation of each item according to some criteria (approach, deployment and results). The points assigned to each item are added up to get an overall assessment for an organization. See the appendix for a detailed description of each category.

European Quality Award

This award suggest a method which assigns percentages according to some criteria (approach and deployment, results and scope) for each category. An overall percentage score is then assigned to each category, which is then converted to a point score. Adding up point scores provides the overall evaluation for the organization. See details in the appendix.

Both methods assign points which indicate the importance of each category. Malcolm Baldrige defines points for categories and items, whereas the European Quality Award defines points only at the category level. Both methods use a percentage scoring system to determine the degree of accomplishment at lower levels of detail. Both methods suggest a point score up to 1000 points.

Using the point method according to the importance of elements in the framework, such as presented in section 3.1, we obtain the following table:

Dimension	Factor	Points
Project		110
	Process	21
	Agent	79
	Tool	10
System		260
	Product	156
	Performance	78
	Technology	26
Environment		630
	Contribution	397
	Compliance	164
	Usability	69

Points are assigned using the same distribution we have in our approach (i.e., .63 for the environment dimension, .26 for the system dimension and .11 for the project dimension). The objective is to compare this method with the framework under similar conditions.

Let us apply these points to the information systems evaluated previously. The method is as follows: for each factor, look up its category from the original information system profile. If the category is basic, apply 17% of the points; if the category is intermediate apply 41% of the points; if the category is advanced apply 100% of the points. These percentages represent the same proportion we used in the hierarchical approach. The following table shows the assignment of points for each factor, and the global evaluation obtained by adding partial results.

Information System Points

	1	2	3	4	5	6	7	8
Process	**9**	**9**	**9**	**9**	**9**	**9**	**9**	**9**
Agents	**79**	**32**	**79**	**32**	**32**	**32**	**79**	**79**
Tools	**4**	**2**	**4**	**4**	**4**	**2**	**2**	**4**
Product	**156**	**64**	**27**	**64**	**156**	**64**	**27**	**27**
Performance	**78**	**32**	**78**	**13**	**78**	**32**	**78**	**78**
Technology	**11**	**11**	**11**	**4**	**26**	**11**	**11**	**11**
Compliance	**67**	**67**	**67**	**67**	**164**	**67**	**67**	**67**
Usability	**28**	**69**	**69**	**28**	**69**	**12**	**28**	**28**
Contribution	**163**	**397**	**163**	**163**	**163**	**397**	**163**	**397**
Global Evaluation	**595**	**683**	**507**	**384**	**701**	**626**	**464**	**700**

The point percentage approach identifies systems 5 and 8 as the highest evaluated, followed by systems 2 and 6. System 4 receives, as before, the lowest evaluation.

Multiple metrics graph and point percentage approaches are capable of identify differences among the evaluated systems. However, the results do not coincide in identifying the higher evaluation. Multiple metrics graphs identify 2 as the highest evaluated system (followed by systems 8 and 5), whereas point percentages identify systems 5 and 8 as the highest evaluated (system 2 follows in the ranking). Both methods consistently identify 4 as the lowest evaluated system.

Comparing the three methods:

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Hierarchical	.53	.73	.49	.33	.60	.70	.43	.80
Multiple Metrics Graph	.62	.85	.54	.33	.71	.61	.49	.77
Point percentage	.60	.68	.51	.38	.70	.63	.46	.70

Comparing these methods to the hierarchical approach, it is possible to establish the sensitivity of the hierarchical appraoch. This approach is much more sensitive when the

aggregated values are assigned, because the importance is allocated between factors. System 8 was the highest evaluated (followed by systems 2 and 6), because its contribution had the higher relative importance and was categorized as advanced. Comparing system 2 and system 8, both are rated advanced on the contribution factor (.96 versus .93 at the environment dimension level), but system 8 was evaluated higher in both the project and system dimensions (.66 versus .28 at the project dimension level and .53 versus .38 at the system dimension level).

Differences in the aggregated results come from the pairwise assignment of importance among factors in our approach. Both multiple metrics graphs and point percentage, provide an absolute score which is not capable of portraying the relative importance among all the factors by level, whereas our approach is built upon these multiple factor comparisons.

5.0 Conclusions, current and further research

We have presented a hierarchical approach to aggregating software systems evaluation data. A software systems evaluation framework which integrates the important dimensions of software systems in a hierarchy has been used as the baseline of the approach. The hierarchical approach represents an original application of measurement which aggregates logically related attributes into metrics, following the levels in the hierarchy. Results of the aggregation are used to characterize systems from a high-level perspective (i.e., managers, users). Using evaluations per level in the hierarchy, it is possible that different points of view can find their required level of information.

The approach can be used for any evaluation hierarchy that requires data aggregation. The approach does not depend on the number of levels or the number of elements in the hierarchy. It permits the identification of hot spots in the evaluation by looking at each level. It provides information that can be used by technicians or managers at the desired level of aggregation. The examples of evaluations demonstrate the potential of the approach, which is to identify similar projects by global evaluation or by similar-dimension evaluation while keeping factor categorization available to accomodate additional detail.

Compared to existing approaches, the hierarchical approach is more sensitive. Multiple metrics graphs provide an evaluation method based on computing the surface in a weighted Kiviat graph. Quality awards use a point and percentage evaluation method which weights each dimension. The difference in the hierarchical approach is the pairwise comparisons between dimensions or factors, a feature not captured by these methods. Another difference is that measurement in the hierarchical approach is based on a ratio scale rather than an absolute scale.

Results indicate the potential benefits of a quantitative method for software systems evaluation. The accuracy of the evaluation is improved as measurement presents results within a range of permissible values (i.e., normalization). The method documents all the steps that assign a value to the relative importance of the elements. Comparisons among evaluations from different organizations can be performed using the same importance matrices.

Each organization can determine its own importance parameters, however practitioners are requesting assistance in assessing the relative importance of factors and dimensions in the framework. Testing the approach in industry is also underway, and several systems are being evaluated using the approach. The evaluation framework is being extended to include several metrics that are important for each dimension or factor. Extending the model to include the degree of importance of participants in the evaluation process is being analyzed.

Further research to automatically evaluate systems from historical metrics data is deemed important. Availability of subjective and objective metrics for each factor in the framework requires cohesive approaches for integration. As the amount of information being gathered for a system increases, mechanisms have to be devised to determine commonality among metric data. The goal would be to automatically assign an evaluation to a system, thus minimizing the need for an evaluator judgment.

Acknowledgements

The authors would like to acknowledge the contributions involving the following people and organizations: Andre Beaucage which provided initial feedback when the CASE tools were evaluated; Helene Beneteau de Laprairie and Jean-Sebastien Neveu for their useful comments. We also thank Groupe DMR which provided real information systems evaluations from industry.

References

[BR94] Boloix, G.; Robillard, P.N. 'Comprehensive Software Metrics Framework', Technical report EPM/RT-94/07, Ecole Polytechnique de Montreal, March 1994.

[CG90] Card, D.; Glass, R. 'Measuring Software Design Quality', Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[EQ94] European Quality Award, Self-assessment based on the European Model for Total Quality Management 1994, The European Foundation for Quality Management, 1994.

[MBA94] Malcolm Baldrige National Quality Award, 1994 Award Criteria, United States Department of Commerce, Technology Administration, National Institute of Standards and Technology, 1994.

[MK90] Munson, J.C.; Khoshgoftaar, T. 'Applications of a relative complexity metric for software project management', Proceedings of the Third Annual Oregon Workshop on Software Metrics, Silver Falls, Oregon, 1990.

[PFR92] Pfleeger, S.L.; Fitzgerald, J.C.; Rippy, D.A. 'Using multiple metrics for analysis of improvement, Software Quality Journal 1, 27-36 (1992).

[S80] Saaty, T.L. 'The Analytic Hierarchy Process', McGraw-Hill Inc., 1980.

Appendix

Software Systems Evaluation Framework

Model description

The objective of the software systems evaluation framework is to provide a mechanism to classify software characteristics consistently. The classification schema would be appropriate for identifying sets of similar projects and identifying the range of projects undertaken by an organization. Let us describe the characteristics of the factors and their corresponding categories for each dimension.

PROJECT

A project may be oriented towards different types of activities (i.e., development or maintenance). A system may be subject to several changes after development (e.g., correction, modification, enhancement) during its lifetime, each change performed through different projects and each project having different characteristics. Project information is relevant during the initial development of the system, and also later on during evolution.

The project dimension seeks to characterize project efficiency considerations (i.e., the ability to develop a system without waste of time, energy, etc.)

Process

The process factor seeks to evaluate the degree of efficiency and continuity of the process. This is primarily oriented towards process management assessment.

Process categories

Basic: The project is characterized as one without a stable environment for producing software. Methodologies are adapted for each project and there is no follow-up of organizational learning from experience. Performance can only be predicted by individual, rather than project, capability.

Intermediate: The project follows a standard process for producing software. The software engineering and software management processes group facilitates software process definition and improvement efforts. Projects use the organization-wide, standard software process to create their own defined software process which encompasses the unique

characteristics of the project. Each project uses a peer review process to enhance product quality.

Advanced: The project sets quantitative quality goals for software products. Productivity and quality are measured for important software process activities. Software processes have been instrumented with well-defined and consistent measures which establish the quantitative foundation for evaluating project processes and products.

Agent

The agent factor seeks to assess the capability of the team of people participating in the project. Management and technical personnel should be included in the assessment .

Agent categories

Basic: The team is unprepared to undertake the project. There is some experience with similar applications, system design issues and programming techniques, but they are insufficient to build the required system. Extra effort is required from participants to improve their competence (e.g., attending training sessions, working overtime).

Intermediate: The team has the capability to undertake the project. Team members have varied levels of experience with related applications, system design and programming techniques. There is some level of confidence in their chances of success.

Advanced: The team has already successfully demonstrated its capacity to undertake similar projects. Team members have a consistent mix of experiences with related applications, system design and programming techniques. There is confidence that successful results will be achieved.

Tool

The tool evaluation factor requires the establishment of the level of sophistication of tool support for development and maintenance activities. It is oriented towards idenifying tools used by technicians rather than managers.

Tool categories

Basic: A basic programming toolkit is available for software production (i.e., compilers, debuggers and testers).

Intermediate: An improved programming toolkit is available for software production. Programming, verification and validation, configuration management and measurement tools are available during software production. The use of workbenches on a limited basis may be available.

Advanced: Software development environments and CASE tools are available for software production which include software and process metrics. User interface development workbenches are used regularly.

SYSTEM

There are several categories of systems widely known in the software community. Some examples are real-time computing, fault-tolerant computing, high-safety and high-security computing, high-performance networks and multimedia technologies.

The system dimension is oriented towards evaluating internal and external product characteristics and the system's implementation technology.

Product

The product factor requires determination of an overall assessment of the internal product characteristics: size, quality and complexity.

Product categories

Basic: Software applications which implement few functions with low complexity and low quality requirements.

Intermediate: Software applications which implement several functions with increasing requirements in terms of quality and complexity.

Advanced: Software applications which implement a large number of functions with high levels of complexity and increasingly stringent quality requirements.

Performance

The performance factor concerns the assessment of external software characteristics: reliability and efficiency. It concerns dynamics rather than static characteristics.

Performance categories

Basic: The system is not bound by reliability and efficiency concerns. The effect of a software failure is an easily recoverable loss for users.

Intermediate: The system requires increased support in terms of reliability and efficiency. The effect of a software failure is a situation from which users can recover without extreme penalty.

Advanced: The system generates major concern in terms of reliability and efficiency requirements. A software failure can mean a serious financial loss or a massive human inconvenience.

Technology

The technology factor requires the establishment of the level of sophistication of the technology implementing the system.

Technology categories

Basic: Single-user and multi-user microcomputers and minicomputers.

Intermediate: Multi-user mainframe technology.

Advanced: Open-system computer technology allowing distributed computing.

It would convenient to identify precisely the language and DBMS implementing the system and the operating system supporting its operation. Any combination of languages may be accessible to implement the system. Operating systems and database management systems are usually linked to specific technologies.

ENVIRONMENT

The environment domain evaluates the user's satisfaction with the system.

Compliance

The compliance factor assesses the degree of accomplishment of the user's requirements.

Compliance categories

Basic: Basic requirements are partially fulfilled by the system. There is no major impact on the users or their jobs. Improvements to the system are required in the short term.

Intermediate: Basic requirements are fulfilled by the system. There is an impact on the users and their jobs. Some areas of the system might be improved, but this is not considered to be urgent.

Advanced: Requirements are fulfilled by the system.The service provided by the system is appropriate, the information is adequate, current and timely. There is a major positive impact on the users and their jobs.

Usability

The usability factor is directed towards assessing user interface characteristics. The impact on learnability is considered an important aspect in evaluating usability.

Usability categories

Basic: Learning the system may require a great deal of time. The system is not providing interfaces in user terms. Redundant information is required from the users.

Intermediate: The system requires some time to learn, but this is considered acceptable by the users. The system provides interfaces in user terms. There are some concerns about the system because error messages are not clear and there are no shortcuts for experienced users.

Advanced: The system requires a short time to learn. The system is user-oriented, efficient to use, prevents errors by the user and clearly signals the seriousness of any errors. Infrequent users have the facility to return to using the system without having to relearn it, and frequent users find shortcuts that improve their efficiency.

Contribution

> The evaluation of the contribution of the system to the organization assesses the benefits provided by the system to the organization

Contribution categories

Basic: There is no major contribution to the organization, the system has automated the same tasks as they were performed manually.

Intermediate: There are some intangible benefits to the organization. Decision-relevant information is processed in a cost-effective way, improving the quality and speed of management decision-making processes.

Advanced: The system provides tangible benefits to the organization. The organization is rendered more cost-effective by using the system. Internal coordination costs have been reduced.

Quality Awards

Malcolm Baldrige National Quality Award

Leadership: The senior executives' success in creating and sustaining a quality culture.

Information and Analysis: The effectiveness of information collection and analysis in maintaining a customer focus, in driving quality excellence and in moving toward excellence.

Strategic Quality Planning: The effectiveness of integrating quality requirements into business plans.

Human Resource Development and Management: The success of efforts to realize the full potential of the work force to meet a company's quality and performance objectives.

Management of Process Quality: The effectiveness of systems and processes in ensuring the quality of products and services.

Quality and Operational Results: The improvement in quality and operational performance and supplier quality, demonstrated through quantitative measures.

Customer Focus and Satisfaction: The effectiveness of systems to determine customer requirements and satisfaction and demonstrated success in meeting customers' expectations.

Points assigned by the award per category

Table 2: Point Values

Categories / Items	Point Values
1.0 Leadership	95
1.1 Senior Executive Leadership	45
1.2 Management for Quality	25
1.3 Public Responsibility and Corporate Citizenship	25
2.0 Information and Analysis	75
2.1 Scope and Management of Quality and Performance Data and Information	15
2.2 Competitive Comparisons and Benchmarking	20
2.3 Analysis and Uses of Company-Level Data	40
3.0 Strategic Quality Planning	60
3.1 Strategic Quality and Company Performance Planning Process	35
3.2 Quality and Performance Plans	25
4.0 Human Resource Development and Management	150
4.1 Human Resource Planning and Management	20
4.2 Employee Involvement	40
4.3 Employee Education and Training	40
4.4 Employee Performance and Recognition	25
4.5 Employee Well-Being and Satisfaction	25
5.0 Management of Process Quality	140

Table 2: Point Values

Categories / Items	Point Values
5.1 Design and Introduction of Quality Products and Services	40
5.2 Process Management: Product and Service Production and Delivery Processes	35
5.3 Process Management: Business and Support Service Processes	30
5.4 Supplier Quality	20
5.5 Quality Assessment	15
6.0 Quality and Operational Results	180
6.1 Product and Service Quality Results	70
6.2 Company Operational Results	50
6.3 Business and Support Service Results	25
6.4 Supplier Quality Results	35
7.0 Customer Focus and Satisfaction	300
7.1 Customer Expectations: Current and Future	35
7.2 Customer Relationship Management	65
7.3 Commitment to Customers	15
7.4 Customer Satisfaction Determination	30
7.5 Customer Satisfaction Results	85
7.6 Customer Satisfaction Comparison	70

The European Quality Award

ENABLERS

1. **Leadership**: The behavior of all managers in driving the organization towards Total Quality

1.a Visible involvement in leading Total Quality

How managers take positive steps to:

- communicate with staff

- act as role models, leading by example

- make themselves accessible and listen to staff

- give and receive training

- demonstrate commitment to Total Quality

1.b A consistent Total Quality culture

How managers take positive steps to:

- be involved in assessing awareness of Total Quality

- be involved in reviewing progress towards Total Quality

- include commitment to and achievement in Total Quality in appraisal and promotion of staff at all levels

1.c Timely recognition and appreciation of the efforts and successes of individuals and teams

How managers are involved in recognition:

- at local, section and group levels

- at divisional level

- at the organization level

- of groups outside the organization e.g., suppliers and customers

1.d Support of Total Quality by provision of appropriate resources and assistance

How managers provide support through:

- helping to define priorities in improvement activities

- funding, learning, facilitation and improvement activities

- actively supporting those taking quality initiatives

- releasing staff to participate in Total Quality activities

1.e Involvement with customers and suppliers

How managers take positive steps to:

-meet, understand and respond to the needs of customers and suppliers

- establish and participate in 'partnership' relationships with customers and suppliers

- establish and participate in joint improvement activities with customers and suppliers

1.f Active promotion of Total Quality outside the organization

How managers promote quality management outside the organization through:

- membership in professional bodies

- publication of booklets, articles

- presentations at conferences, seminars

- support of local community

2. **Policy and Strategy**: The organization's mission, values and strategic direction and the manner in which it achieves them

2.a How policy and strategy are based on the concept of Total Quality

2.b How policy and strategy are formed on the basis of information that is relevant to Total Quality

2.c How policy and strategy form the basis for business plans

2.d How policy and strategy are communicated

2.e How policy and strategy are regularly reviewed and improved

3. **People Management**: The management of the organization's people

3.a How continuous improvement in 'people management' is accomplished

3.b How the skills and capabilities of the people are preserved and developed through recruitment, training and career progression

3.c How people and teams agree on targets and continuously review performance

3.d How the involvement of everyone in continuous improvements is promoted and people are empowered to take appropriate action

3.e How effective top-down and bottom-up communication is achieved

4. **Resources**: The management, utilization and preservation of resources

4.a Financial resources

4.b Information resources

4.c Material resources and fixed assets

4.d The application of technology

5. **Processes**: The management of all value-adding activities within the organization

5.a How processes critical to the success of the business are identified

5.b How the organization systematically manages its processes

5.c How process performance measurements, along with all relevant feedback, are used to review processes and to set targets for improvement

5.d How the organization stimulates innovation and creativity in process improvement

5.e How the organization implements process changes and evaluates the benefits

RESULTS

6. **Customer Satisfaction**: What the perception of your external customers is of the organization and of its products and services

7. **People Satisfaction**: What your people's feelings are about their organization

8. **Impact on Society**: What the perception of your company is in the community at large. This includes views of the company's approach to quality of life, the environment and to the preservation of global resources

9. **Business Results**: What the organization is achieving in relation to its planned business performance

Table 3: Point Values

Elements	%
Leadership	10
People Management	9
Policy and Strategy	8
Resources	9
Processes	14
People Satisfaction	9
Customer Satisfaction	20
Impact on Society	6
Business Results	15